ARTIFICIAL INTELLIGENCE EXTERNSHIP Project Report

**AUTOMATED ESSAY SCORING USING LSTM**

By

*Team Peregene*

Ashutosh Pandey

Jithaamithran S.

Pavan Kumar A.

Tanisha Mandal

# **Table of Contents**

1. Introduction 2
2. Problem Statement 2
3. Solution 3
4. Literature Survey 6
5. Experimental Investigation 7
6. Software Specifications 7
7. Flowchart 11
8. Conclusion 11
9. Future Scope 12
10. Bibliography 12

**1. Introduction**

Essays are a widely used tool to assess the capabilities of a candidate for a job or an educational institution. Writing an essay given a prompt requires comprehension of a given prompt, followed by an analysis or argumentation of viewpoints expressed in the prompt, depending on the needs of the testing authority. They give a deep insight into the reasoning abilities and thought processes of the author, and hence are an integral part of standardized tests like the SAT, TOEFL and GMAT.

With essays comes the need for personnel qualified enough to carry out the process of grading the essays appropriately and ranking them on the basis of various testing criteria. Our project aims to automate this process of grading the essays with the aid of Deep learning, in particular, using Long Short Term Memory networks which is a special kind of RNNs.

Automated Essay Scoring(AES) allows the instructor to assign scores easily to the participants with a pre-trained deep learning model. This model is trained in such a way that the scores assigned are in agreement with previous scoring patterns of the instructor. So this needs the dataset which contains the information of scores given by the instructor previously. AES uses Natural Language processing, a branch of artificial intelligence enabling the trained model to understand and interpret human language, to assess essays written in human language.

**2.Problem Statement**

**2.1 Problem Definition**

Given the growing number of candidates applying for standardized tests every year, finding a proportionate number of personnel to grade the essay component of these tests is an arduous task. These personnel must be skilled and capable of analysing essays, scoring them according to the requirements of the institution and be able to discern between the good and the excellent.

In addition to this, there are a lot of time constraints in grading multiple essays. This can prove to be cumbersome for a limited number of human essay graders. Having to grade several essays within a deadline can compromise on the quality of grading done. Thus, there is a clear need to automate this process so that the institution carrying out the grading can focus on evaluating other aspects of the candidate's profile.

The challenge was to create a web application to take in the essay and predict a score. We need to train a neural network model to predict the score of the essay in accordance to the rater. The model is to be made using LSTM.

**2.2 Dataset:**

We used the dataset provided by Hewlett Foundation in 2012 for the Automated Student Assessment Prize competition on Kaggle. The characteristics of the dataset we have are:

1. There are 8 different sets of essays, each generated from a single prompt.

2. The average length of the essays is between 150 and 550 word per response

3. Each essay was handed graded by 2 or 3 instructors

4. There are a total of 12978 essays.

**3. Solution**

**3.1 Approach**

In order to meet the need for automation of essay grading, we propose an application that provides an interface for users to choose an essay prompt of their choice and provide a response for the same. The user’s response is graded by the application within seconds and a score is displayed.

This application makes use of the technologies of Natural Language Processing that performs operations on textual input, and LSTM, which is used to train a model on how to grade essays. The application also uses Word2Vec embedding technique to convert the essay into vector so that the model can be trainedIt addresses the issue of time constraints; automated grading takes place within seconds as compared to physical grading which requires minutes per essay. The net amount of time saved over a period of consistently using the application is vast; costs of maintaining human graders are also saved upon.

The application gives output from the pre-trained LSTM model. The model is trained using a dataset provided by Hewlett Foundation in 2012 for a competition on Kaggle.

*3.1.1 Long Short Term Memory*

LSTM is a model that can be used for solving Univariate and Multivariate time series forecasting problems. LSTM is used to learn from the series of past observations to predict the next value in the sequence. It has the ability to learn the context required to make predictions, rather than having this context pre-specified and fixed. The model score shows the mean of the actual value and the predicted value. The model score should be as small as possible.

*3.1.2 Word2Vec Model*

Word2vec is a technique/model to produce word embedding for better word representation. It is a natural language processing method that captures a large number of precise syntactic and semantic word relationships. It is a shallow two-layered neural network that can detect synonymous words and suggest additional words for partial sentences once it is trained. Word2vec is a two-layer network where there is input, one hidden layer and output. Word2vec represents words in vector space representation. Words are represented in the form of vectors and placement is done in such a way that similar meaning words appear together and dissimilar words are located far away. This is also termed as a semantic relationship. Neural networks do not understand text, instead they understand only numbers. Word Embedding provides a way to convert text to a numeric vectorWord2vec learns vector representation of words through the contexts.

**3.2 Algorithm**

This is a brief idea of how the model training works

1. Pre-process the data set and extract independent variable x and dependent variable y. Here the independent variable is every essay in the essay column and the dependent variable is domain1\_score.

2. Apply 5-fold cross validation to the x and y variable.

For each fold of cross validation,

a. Split the x and y into x\_train, x\_test and y\_train and y\_test respectively

b. Train the word2vec model for each word tokenized essay in x\_train with parameters set as given in table 1.

c. Generate test and train data vectors using trained word2vec model

d. Train the LSTM model using the reshaped train data vector in the numpy array and y\_train with a batch size of 64 and 150 epochs.

e. Predict y\_pred using test data vector and round off to nearest values.

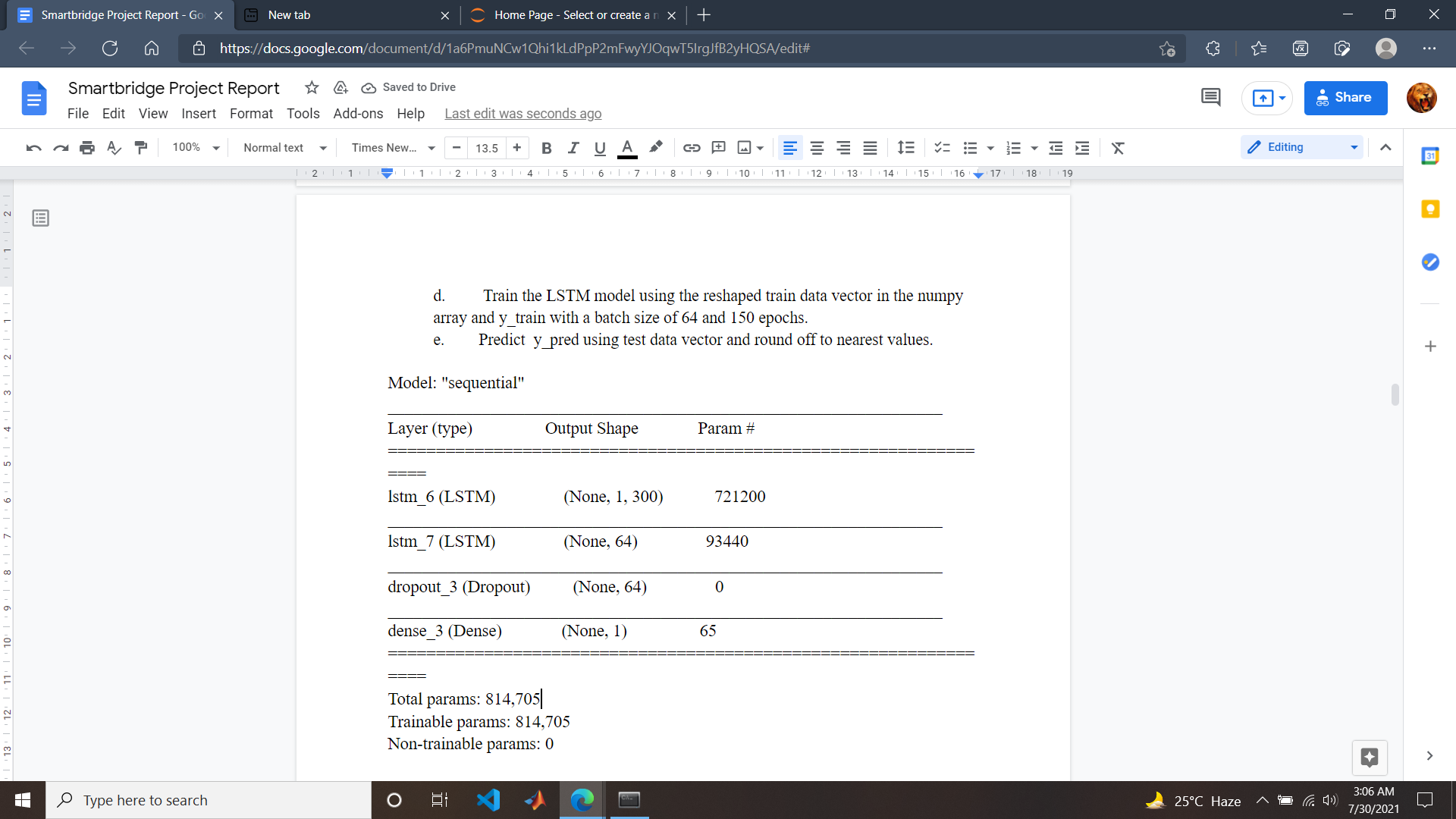


Figure 1 Sequential Mode Parameters for each fold of cross validation

| *Parameter* | *Value* |
| --- | --- |
| num\_features/vector\_size | 300 |
| min\_word\_count | 40 |
| num\_workers | 4 |
| window/context | 10 |
| sample | 1e-3 |

Table 1. Word2Vec Model training parameters.

When the user gives the input essay in the HTML page, flask renders the essay into the program and processes it into data vectors using the saved word2vec model which was previously trained. Then it passes it to the model and subsequently gets the required score and prints it on the HTML page for the user to see the score.

**4. Literature Surveys**

#### [**Valenti, Neri & Cucchiarelli, 2017**](https://doi.org/10.28945%2F331)**. Bayesian Essay Test Scoring System™ (BETSY)**

#### BETSY classifies the text of an essay to a set of groups (Pass-Fail) and (Advanced - Proficient - Basic - Below Basic). BETSY needs to be trained on a huge number (1,000 texts) of human classified essays to learn how to classify new essays. It has been designed to automate essay scoring, but can be applied to any text classification task.

#### [**Alikaniotis, Yannakoudakis & Rei, 2016**](https://doi.org/10.18653%2Fv1%2FP16-1068)**;** [**Taghipour & Ng, 2016**](https://doi.org/10.18653%2Fv1%2Fd16-1193)**. Automatic text scoring using neural networks**

A deep neural network model capable of learning features automatically to score essays was developed. This model has introduced a novel method to identify the more discriminative regions of the text using: (1) a Score-Specific Word Embedding (SSWE) to represent words and (2) a two-layer Bidirectional Long-Short-Term Memory (LSTM) network to learn essay representations.

#### [**Taghipour & Ng, 2016**](https://doi.org/10.18653%2Fv1%2Fd16-1193)**. A neural network approach to automated essay scoring**

Recurrent Neural Networks (RNNs) were used to automatically learn the relation between an essay and its grade. Since the system is based on RNNs, it can use non-linear neural layers to identify complex patterns in the data and learn them, and encode all the information required for essay evaluation and scoring

#### [**Dasgupta et al., 2018**](http://aclweb.org/anthology/W18-3713)**;** [**Dong & Zhang, 2016**](https://doi.org/10.18653%2Fv1%2Fd16-1115)**. Automatic features for essay scoring—an empirical study**

This was an empirical study to examine a neural network method to learn syntactic and semantic characteristics automatically for AES, without the need for external pre-processing. A hierarchical Convolutional Neural Network (CNN) structure was built with two levels in order to model sentences separately

#### [**Dasgupta et al. (2018)**](http://aclweb.org/anthology/W18-3713)**. Augmenting textual qualitative features in deep convolutional recurrent neural network for automatic essay scoring**

#### Convolution Recurrent Neural Network architecture was used to score essays automatically. The model considers both word- and sentence-level representations.

#### **5. Experimental Investigations**

**5.1 Dataset inspection**

The dataset contains eight essay sets. Each of the sets of essays was generated from a single prompt. Each essay contains upto 550 words per response.

1. essay\_id: Each essay has an unique id.
2. essay\_set: Number specifies to which set it belongs
3. **essay: English essay written by the student - independent variable**
4. rater1\_domain1: Score given by rater 1 of domain 1 - all essays have this
5. rater2\_domain1: Score given by rater 2 of domain 1; all essays have this
6. rater3\_domain1: Score given by rater 3 of domain 1; few essays from set 8
7. **domain1\_score: Domain 1’s resolved score; all essays have this - output/dependent variable**
8. rater1\_domain2: Rater 1’s domain 2 score; Set2 essays only
9. rater2\_domain2: Rater 2’s domain 2 score; Set2 essays only
10. domain2\_score: Resolved score between the raters; only essays in set 2 have this rater1\_trait1 score — rater3\_trait6 score: trait scores for sets 7–8

By the inspection of the data, we can see that personally identifying information has been replaced by @CAPS1, @CAPS2, @NUM1, @PERSON1 etc.

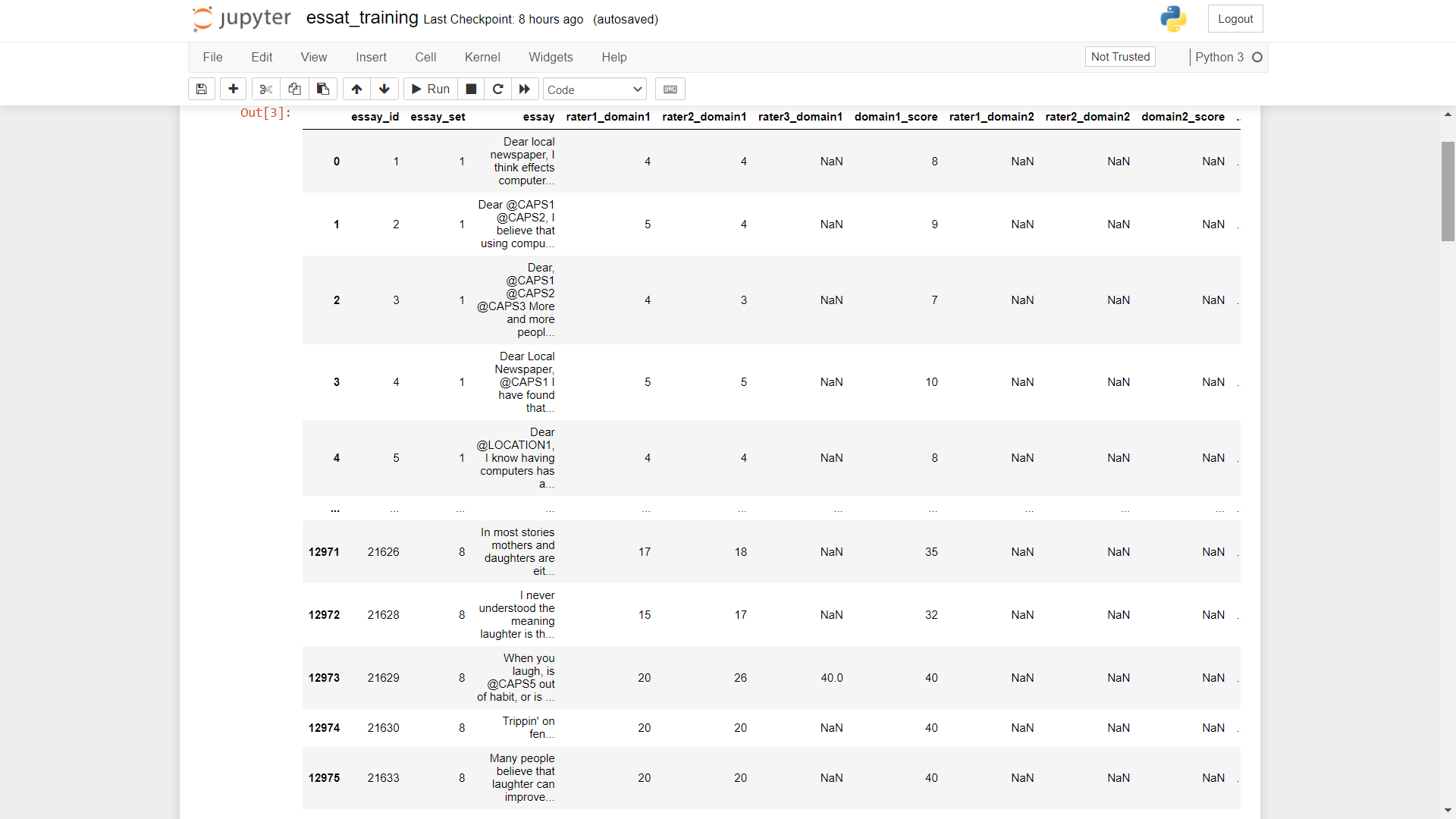


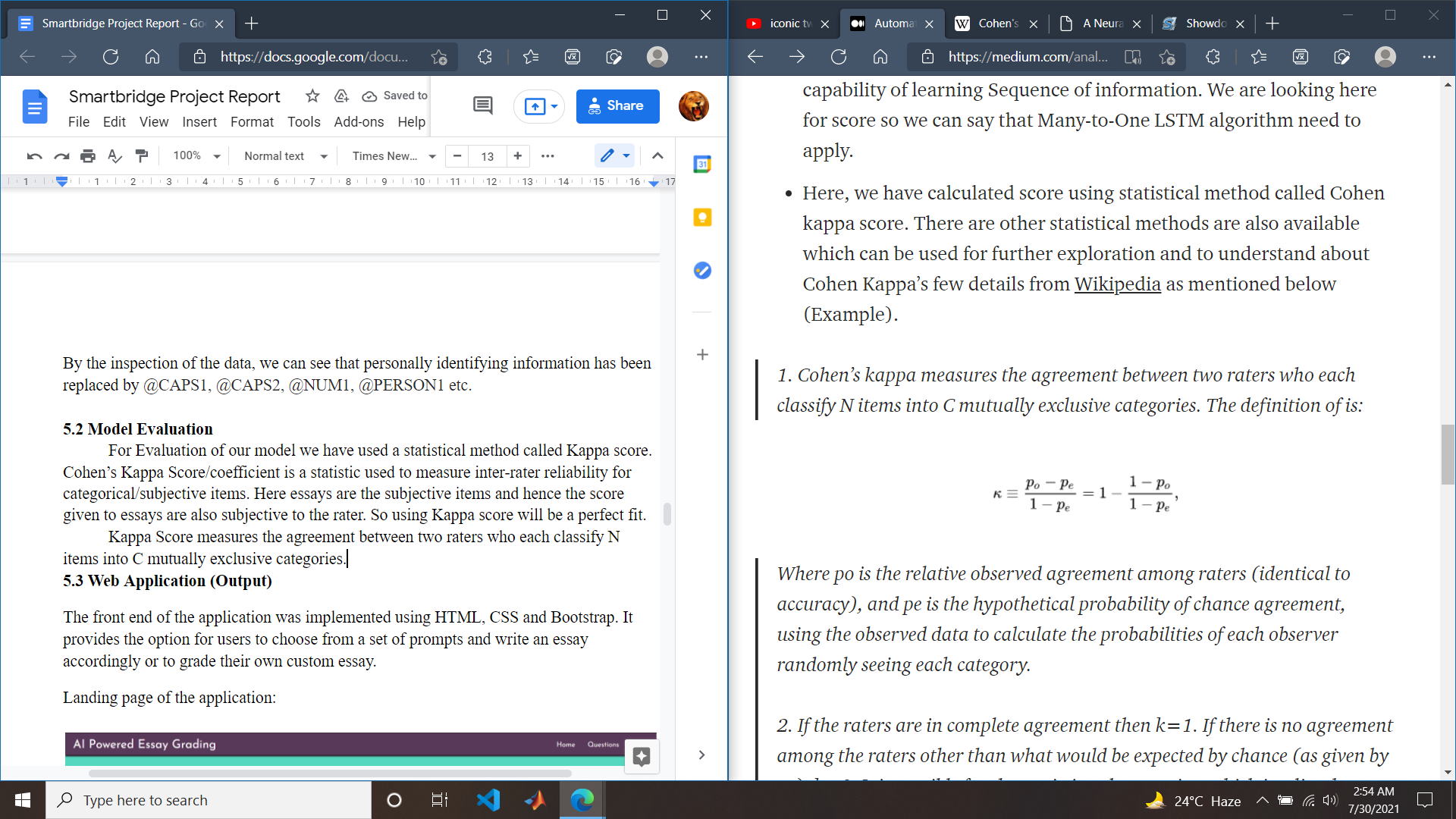
Figure 2 Dataset provided by Hewlett Foundation

**5.2 Model Evaluation**

For Evaluation of our model we have used a statistical method called Kappa score.

Cohen’s Kappa Score/coefficient is a statistic used to measure inter-rater reliability for categorical/subjective items. Here essays are the subjective items and hence the score given to essays are also subjective to the rater. So using Kappa score will be a perfect fit.

Kappa Score measures the agreement between two raters who each classify N items into C mutually exclusive categories.



Where po is the relative observed agreement among raters (identical to accuracy), and pe is the hypothetical probability of chance agreement, using the observed data to calculate the probabilities of each observer randomly seeing each category.

If the raters are in complete agreement then k=1. If there is no agreement among the raters other than what would be expected by chance (as given by pe), k=0. It is possible for the statistic to be negative, which implies that there is no effective agreement between the two raters or the agreement is worse than random.

During 5-Fold Cross Validation, for each fold kappa score and Mean absolute error was measured and noted in the table 2

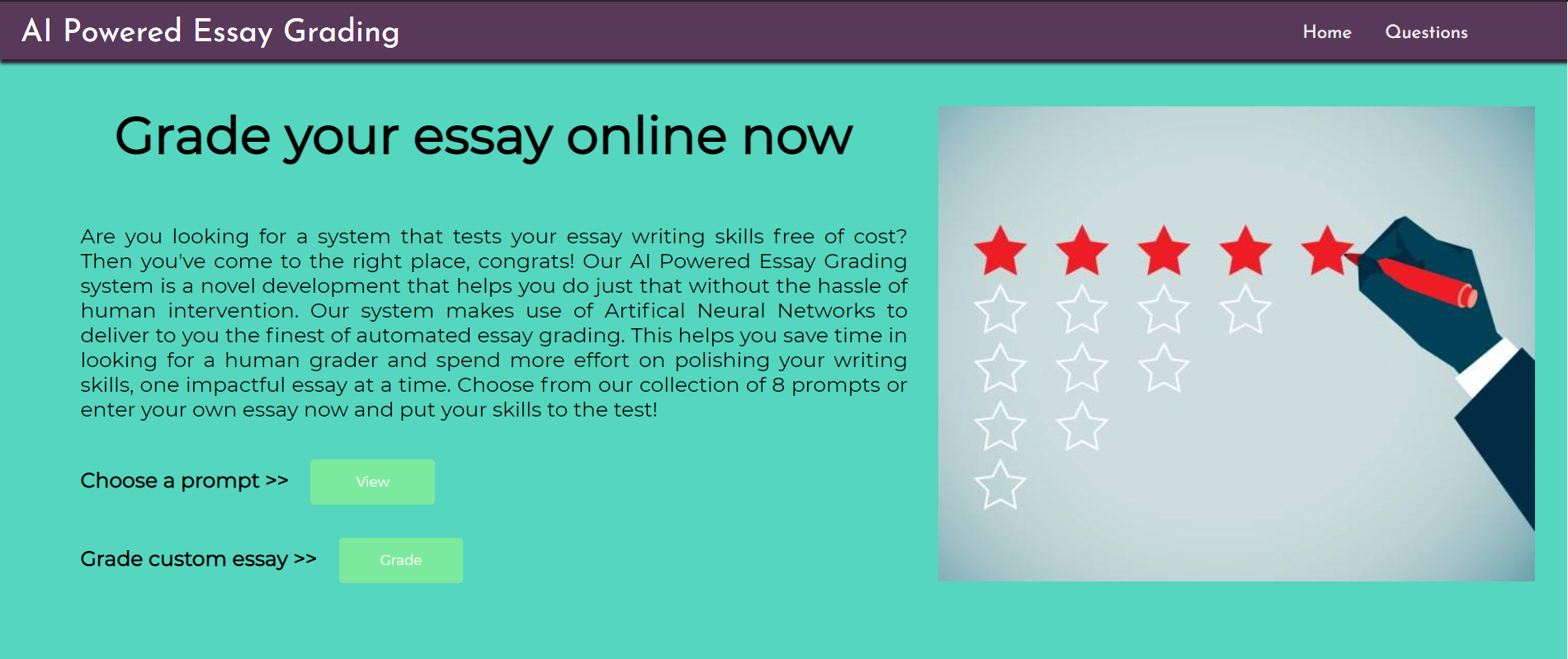
| Nth Fold | Kappa Score | Mean absolute Error | Batch size | Epochs |
| --- | --- | --- | --- | --- |
| 1 | 0.9636 | 1.4291 | 64 | 150 |
| 2 | 0.9630 | 1.4065 | 64 | 150 |
| 3 | 0.9630 | 1.4167 | 64 | 150 |
| 4 | 0.9703 | 1.4061 | 64 | 150 |
| 5 | 0.9649 | 1.4151 | 64 | 150 |

Table 2 Kappa Score and Mean Absolute Error for 5-Fold Cross Validation

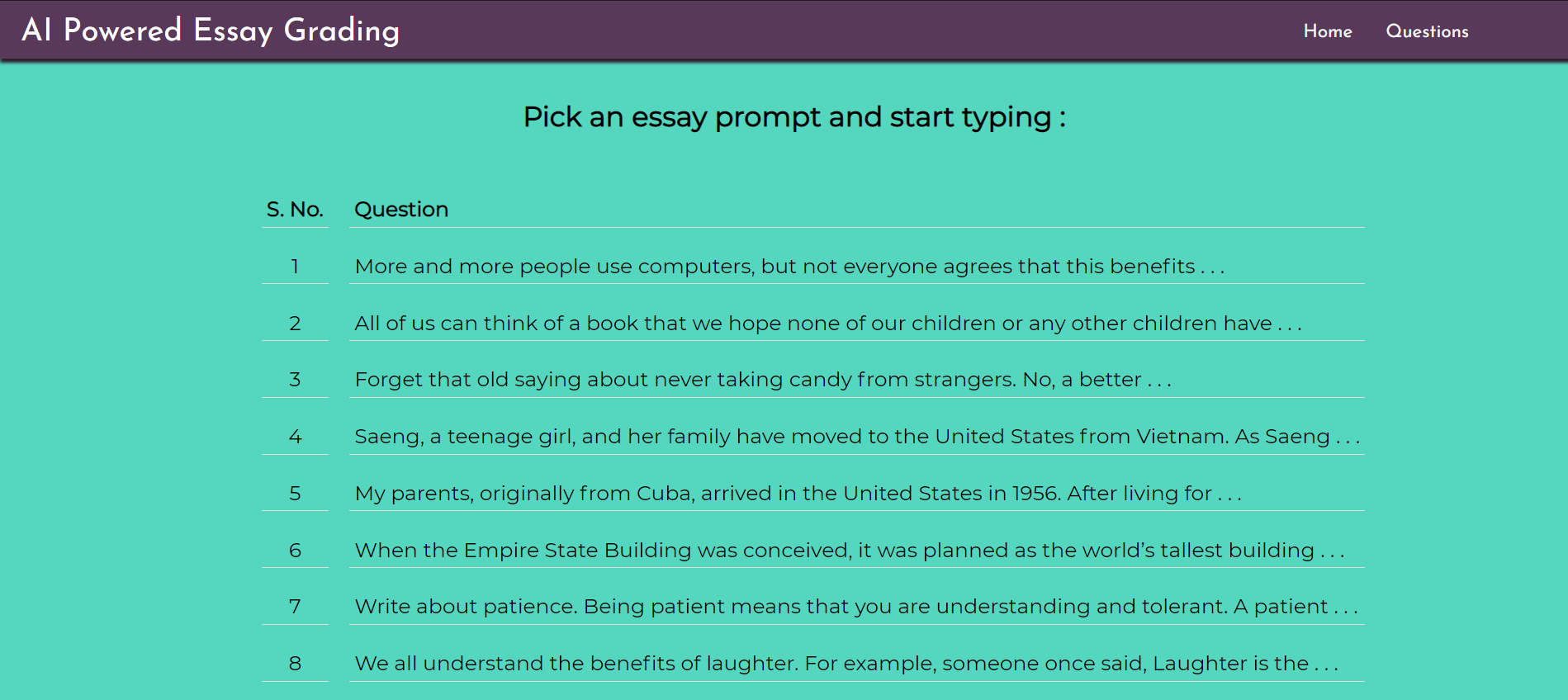
**5.3 Web Application (Output)**

#### The front end of the application was implemented using HTML, CSS and Bootstrap. It provides the option for users to choose from a set of prompts and write an essay accordingly or to grade their own custom essay.

#### Landing page of the application:



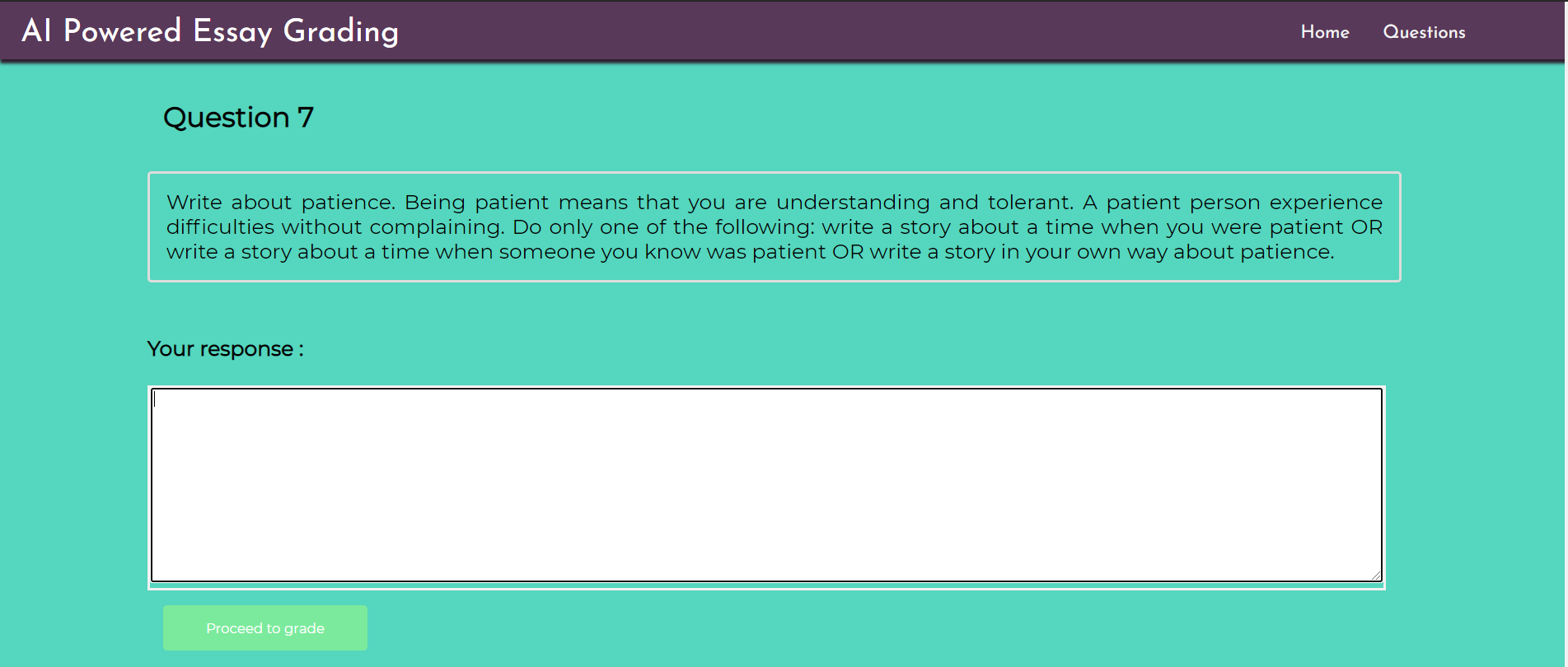
#### Sample essay prompts:



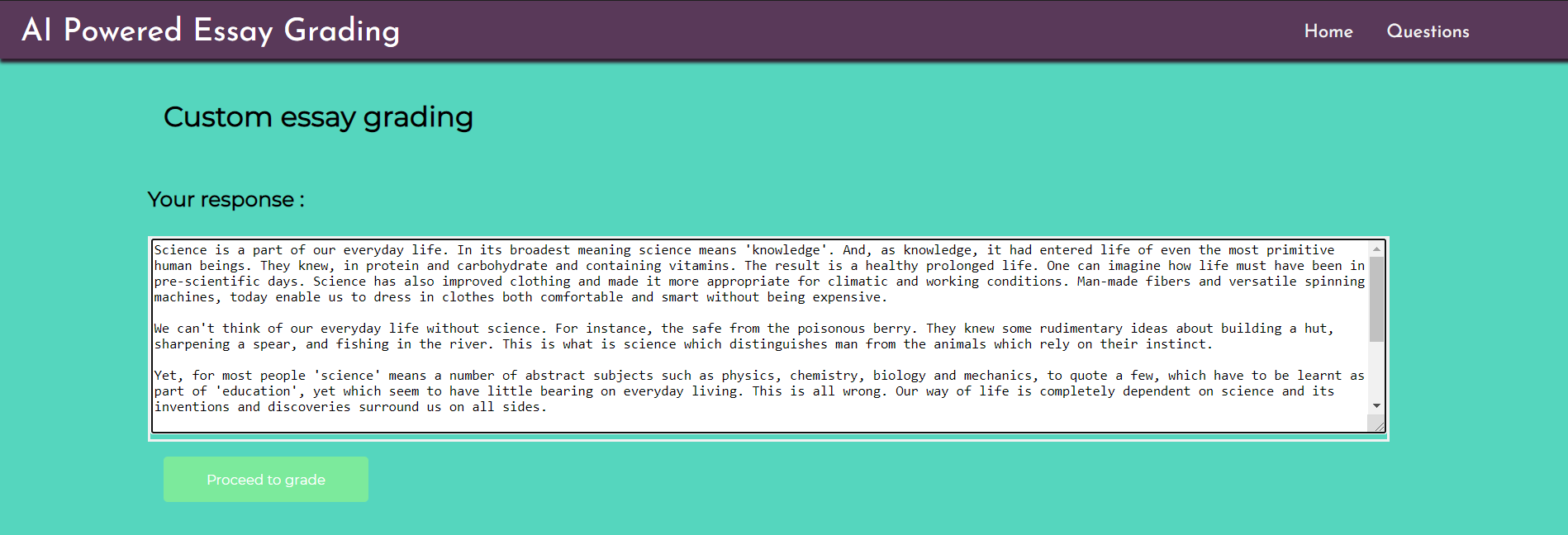
#### 

#### 

#### On clicking on a given prompt:



#### On choosing to grade a custom essay:



#### Final score of user submitted essay:

#### 

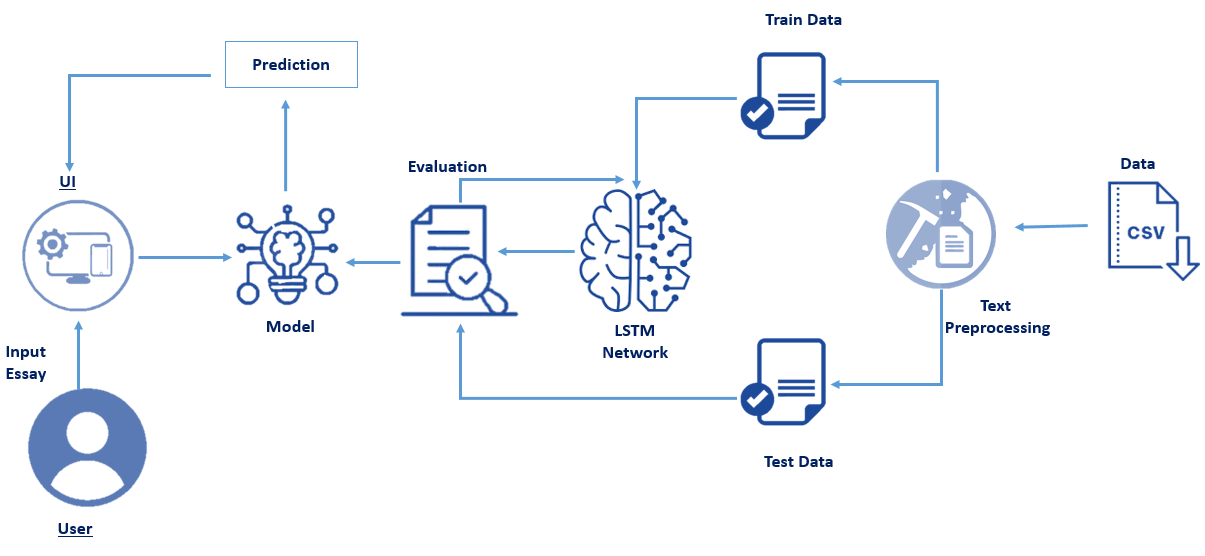
#### **6. Software Specifications**

This application is developed primarily using Python, for the purposes of running the app. The model was built and trained on Jupyter Notebook. The front end of the application was designed with HTML, CSS and Bootstrap. All the components of this application were integrated with the help of Flask App, and the final project was deployed on IBM Cloud.

While training the model, the dataset was imported into the model with Pandas library. Pandas library used was v1.3.0. Numpy v1.19.2 was used to handle array data structure. Natural Language ToolKit v3.6.2 was used to tokenize essays to sentences written in english and also to remove stopwords to make sure the sentences contain only relevant words. RegEx(re) package v2.2.1 was used to remove unnecessary punctuations and symbols present in the essay or sentences. Our model utilizes Word2Vec technique to convert words to corresponding vectors. Word2Vec v0.11.1 was used to convert words into vectors. Tensorflow v2.5.0 was used to build the model. ScikitLearn v0.24.2 was used for data preprocessing.

To make use of the application, the user needs to have access to a stable internet connection and an operating system compatible with the latest versions of most browsers. In the absence of an internet connection, the application can be run locally but the user needs to have authorization to access the source code of our project for the same, which is not recommended for intellectual property purposes.

**7. Flowchart**

****

**8. Future Scope**

This application could be integrated and used by several testing institutions to meet their needs of essay grading. The model used could be trained with an increasing number of input essays to further improve upon its accuracy. The model could also be trained on giving a score on specific criteria of essay grading such as relevancy, linguistic and reasoning ability of the author. Research could be conducted on making the model faster. This technology could also be extended for use with languages other than the English language, effectively rendering it useful on a worldwide level.

**9. Bibliography**

1. Mayur Pethani, Automated Essay Scoring: Kaggle Competition — End to End Project Implementation.

2. Taghipour & Ng, 2016. A neural network approach to automated essay scoring

3. Dasgupta et al., 2018; Dong & Zhang, 2016. Automatic features for essay scoring—an empirical study

4. Dasgupta et al. (2018). Augmenting textual qualitative features in deep convolutional recurrent neural network for automatic essay scoring

5. Automated language essay scoring systems, https://peerj.com/articles/cs-208

6. Valenti, Neri & Cucchiarelli, 2017. Bayesian Essay Test Scoring System™ (BETSY)

7. Essay Writing - Samples, https://www.time4education.com/Bankhomepage-content/Essay-writing-samples.asp

8. Kaveh Taghipour and Hwee Tou Ng, A Neural Approach to Automated Essay Scoring

9. Automatic Text Scoring Using Neural Networks, Dimitrios Alikaniotis, Helen Yannakoudakis, Marek Rei.

10. Alikaniotis, Yannakoudakis & Rei, 2016; Taghipour & Ng, 2016. Automatic text scoring using neural networks